P2P Live Streaming: successes and limitations

Yong Liu

ECE, Polytechnic U.

04/27/2007

joint work with

Keith Ross, Xiaojun Hei, Rakesh Kumar, Chao Liang, Jian Liang

Polytechnic UNIVERSITY Celebrating 150 Years in Brooklyn

Next Disruptive Application?

- Broadband Residential Access
  - Cable/DSL/Fiber to Home
  - BitTorrent, Skype
- Need for Video-over-IP
  - youtube, "video blog"
    - 45 Tera-bytes video, 1.73 billion views -> 1.6billion $
  - video conferencing
  - IPTV
    - live streaming v.s. video-on-demand
    - CNN breaking news v.s. broadcast World of Warcraft
- Impact on Access/Backbone networks

Possible Architectures

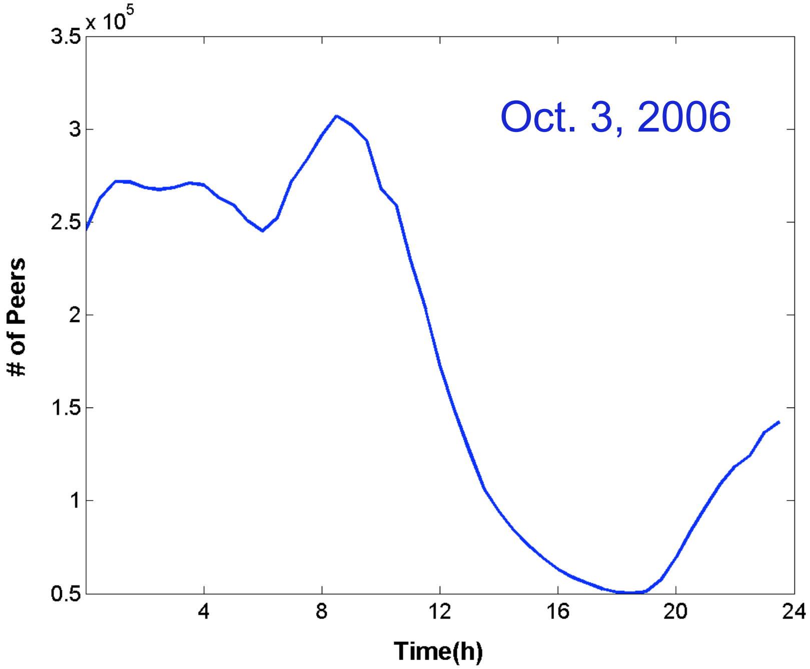
- Native IP Multicast (future Internet?)
- Content Distribution Networks (Youtube)
- Peer-to-Peer Streaming
  - exploit peer uploading/buffering capacity, low cost
  - Push, tree-based designs
    - e.g., end-system multicast from CMU
  - Pull, meshed-based designs
    - inspired by BitTorrent file sharing
    - but with live streaming
    - Coolstreaming, PPLive, PPStream, UUSee, ……

P2P Streaming Success Stories

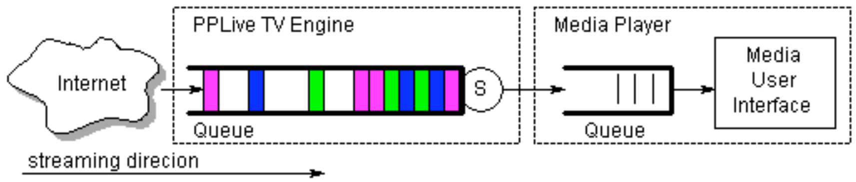
- Coolstream: 4,000 simultaneous users in 2003
- PPLive:
  - 200,000+ users at 400-800 kbps for 4-hours event, 2006 Chinese New Year, aggregate rate of 100 Gbps
  - 400+ channels up to now
    - news, sports, movies, games, special events …

PPLive Overview

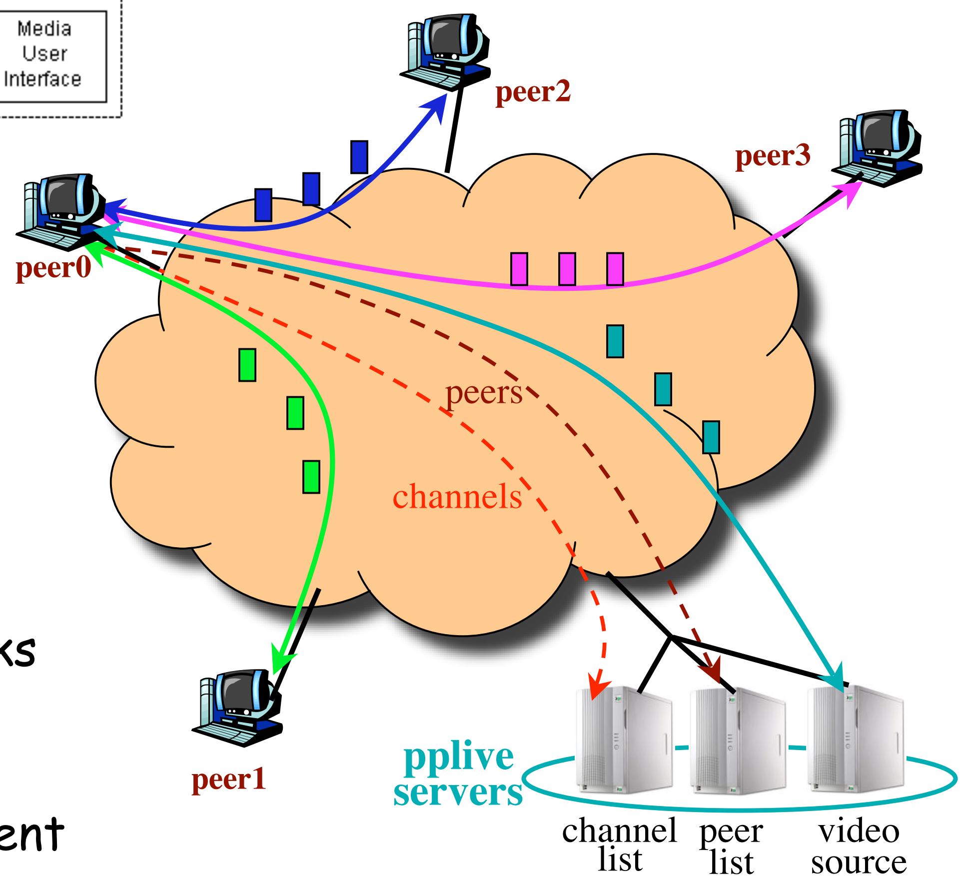
- Free p2p streaming software
  - windows platform, proprietary
  - out of a Univ., China, commercialized
  - popular in Chinese communities since 2005
- 400+ channels, 300K+ users daily
- Video encoded in WMV, RMVB, 300~800kbps
- http://www.pplive.com/



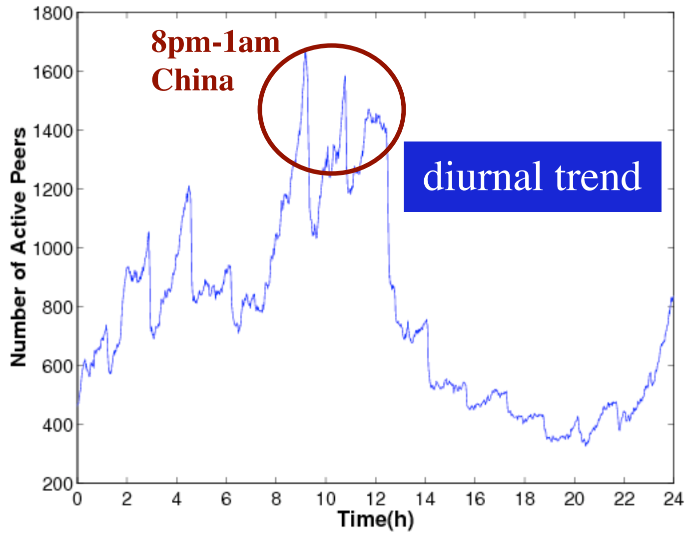
How PPLive works

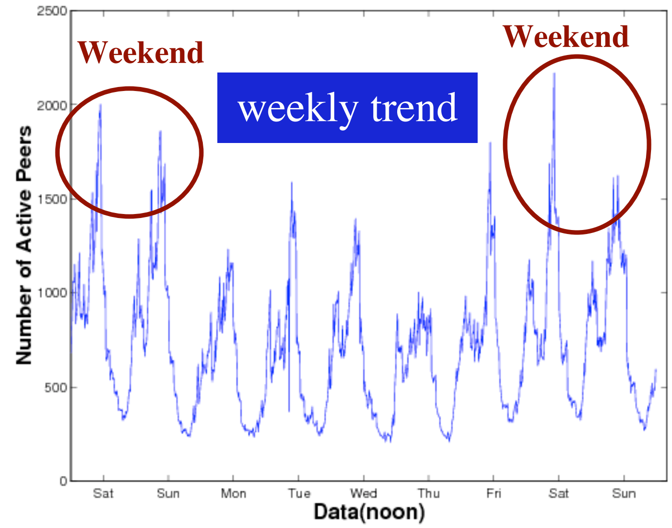


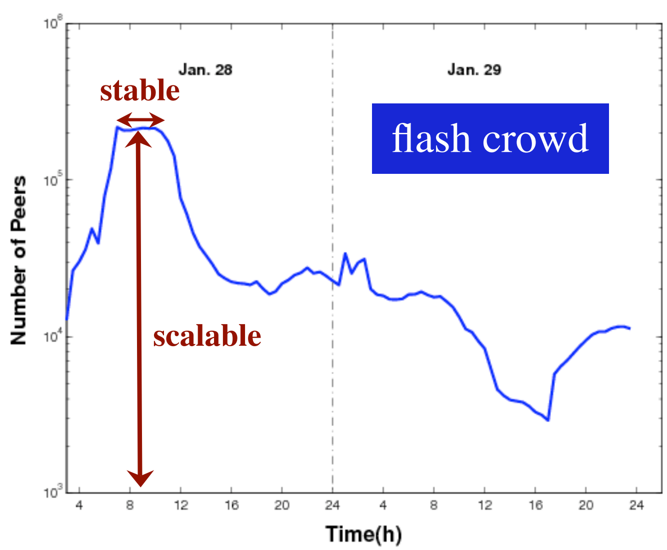
- Signaling not encrypted, protocol analysis through passive sniffing
- BT-Like chunk-driven P2P Streaming
  - register with index server
  - download/upload video chunks from/to peers watching the same channel (TCP)
  - stream buffered video content locally to ordinary media players

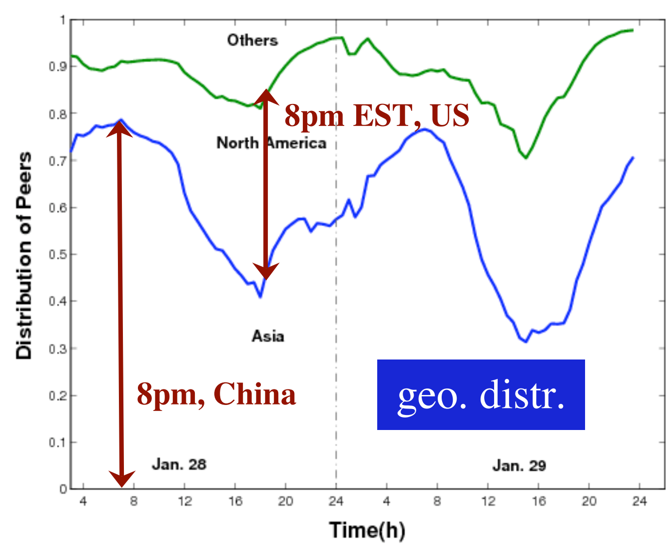


Macro-Stat.: user load









Video Playback Quality

- indirect/unscientific measures
  - subjective feedbacks from users
  - stability of user population (more patient if free?)
  - more peers, shorter delay, fewer freezing, faster recovery
- direct/quantitative measures:
  - start-up delay: 10sec.-3min, “pseudo-realtime”
  - buffer size: 10-30MB
  - playback monitor on local peers
  - buffer map analysis for remote peers

Challenges

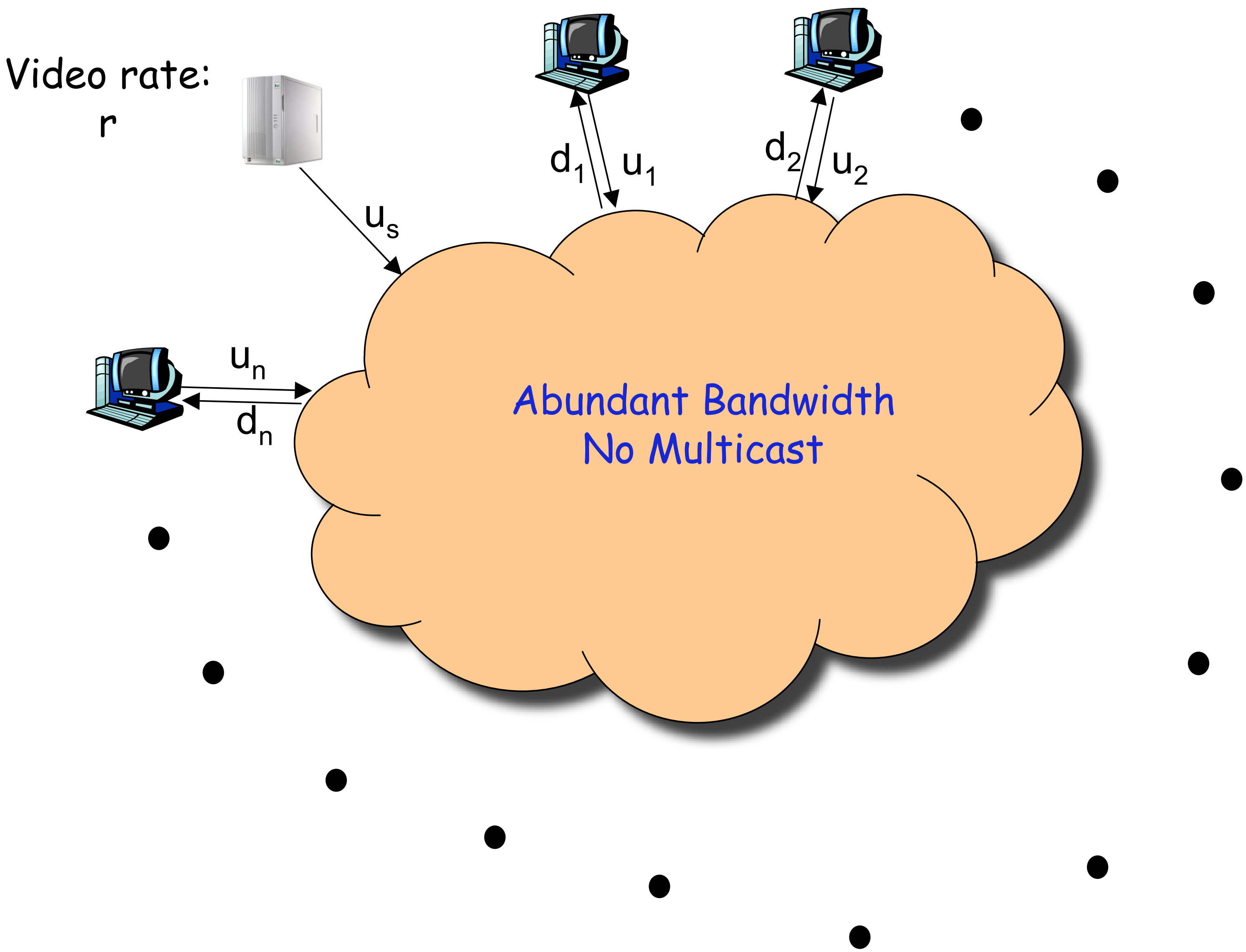
- Bandwidth intensive
  - incentives for redistribution: tit-for-tat?
  - stresses on ISPs
- Asymmetric residential access
  - cable, DSL: upload < download
  - heavily relying on super-peers, e.g., campus nodes
- Peer churn: peers come and go
  - video playback continuity
- Lags among viewers
  - a neighbor cheering for a soccer goal 30 sec.s before you?

Theory

Goal: Expose fundamental characteristics and limitations of P2P streaming systems

- Churnless model (deterministic)
- Churn model

Churnless Model



Maximum video rate r_{max} ?

universal streaming: all peers receive at same rate

$$r_{\max} \le u_s \qquad \text{(rate of fresh content from server)}$$

$$r_{\max} \le d_{\min} \qquad \text{(cannot overwhelm slowest peer)}$$

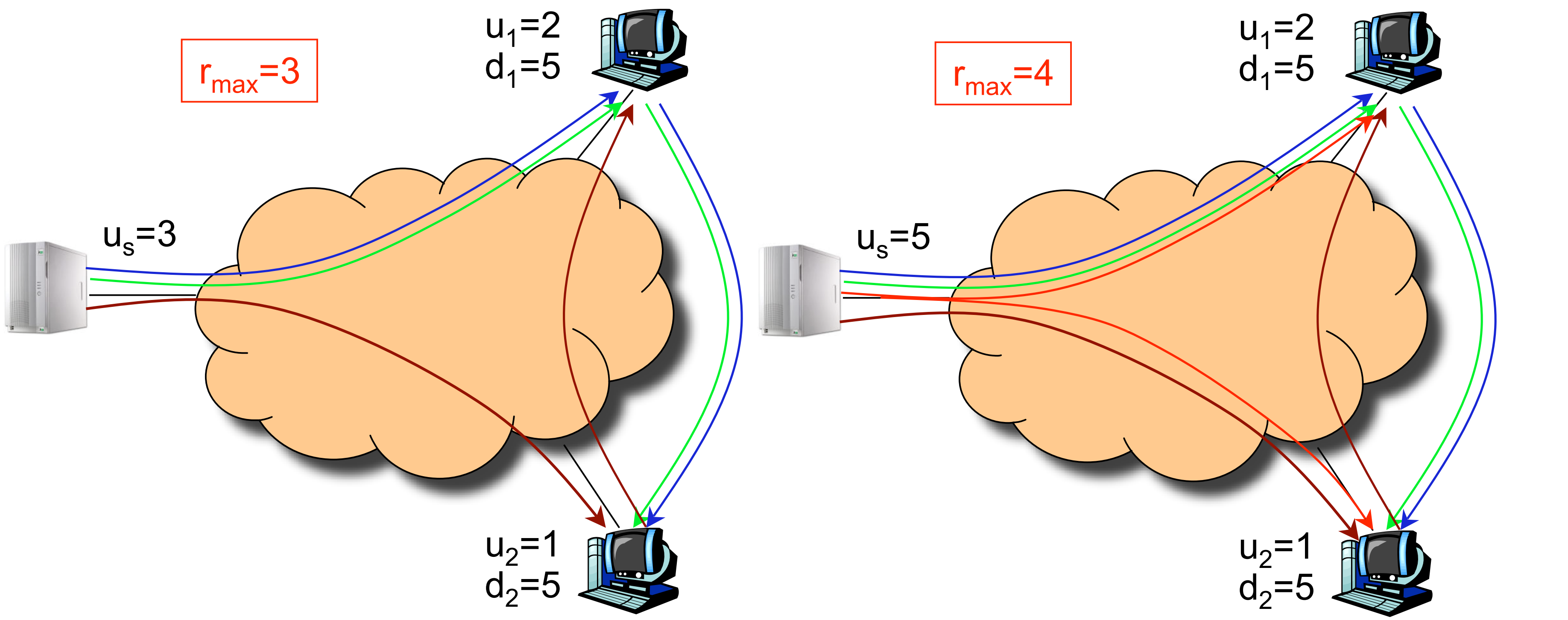
$$r_{\max} \le \frac{u_s + \sum_{i=1}^{n} u_i}{n} \qquad \text{(b.w. demand} \le \text{b.w. supply)}$$

$$\boxed{r_{\max} \stackrel{?}{=} \min\{u_s, d_{\min}, \frac{u_s + \sum_{i=1}^{n} u_i}{n}\}}$$

Theorem: there exists a perfect scheduling among peers such that all peers' uploading bandwidth can be employed to achieve the maximum streaming rate

Perfect Scheduling

- To fully utilize peers' uploading capacity
- Peers with better access upload more



For any peer b.w. dist., two-hop streaming relay achieves maximum rate

Imperfect Internet

- bandwidth sharing
  - among applications on same computer
  - among users in same access
  - congested bottle-neck inside core?
  - → imperfect b.w. info.
  - → rate variations on sessions
- peer churn
  - peers come and go
  - → against static scheduling (tree based)
  - → temporary deficits in uploading capacity
- impact of peer churn, solutions?
  - infrastructural servers
  - peer buffers

Peer Churn Model

- Two peer classes:
  - type 1 ordinary: residential access
  - type 2 super: campus/corporate access
- Upload rate for class i: u_i $u_2 \leq r \leq u_1$
- Arrival rate for class i: η_i
- Average viewing time: $1/\mu_i$
- Li = # of type i, (random variable), $\rho_i = E[L_i]=\eta_i/\mu_i$
- P("universal streaming") = $P(L_1 \geq cL_2 - u')$

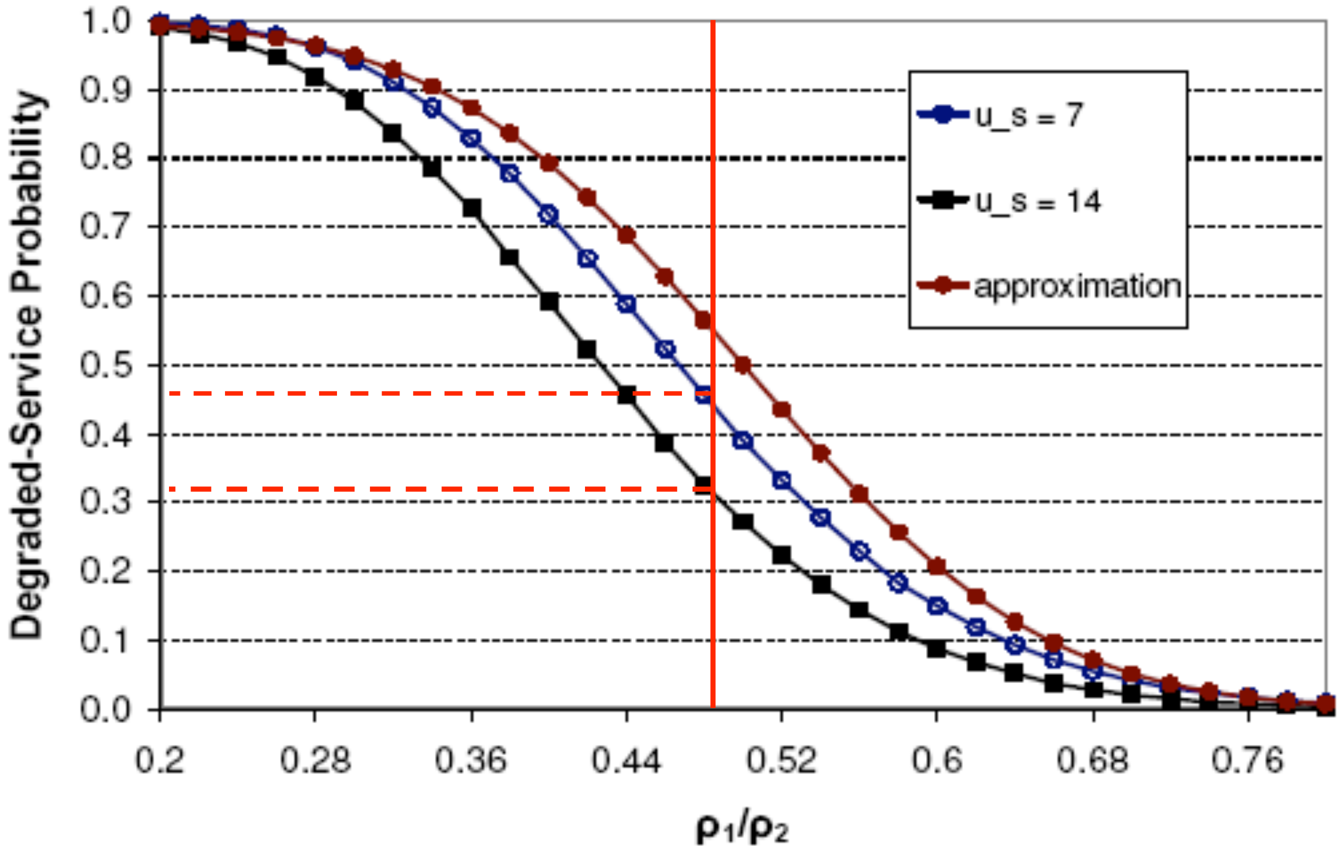
Large System Analysis

- Let ρ_1 and ρ_2 approach ∞
- But ratio $\rho_1/\rho_2 = K$
- More generally

Theorem: In limit, P("univ streaming") =

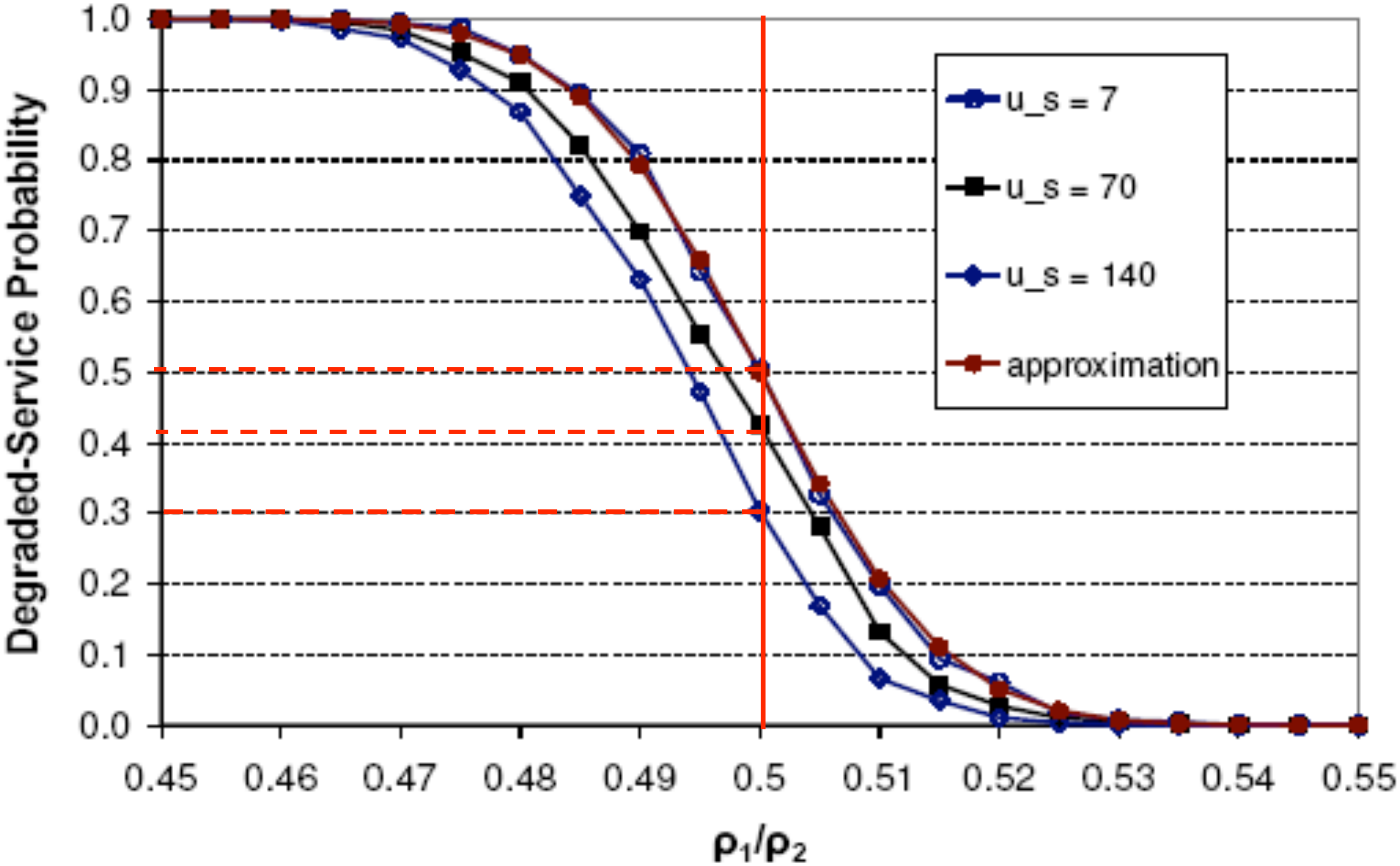
$$\begin{cases} 1 & \text{if} \quad K>c \\ 0 & \text{if} \quad K<c \\ F\left(\dfrac{-\beta}{\sqrt{c+c^2}}\right) & \text{if} \quad K=c \quad \rho_1 = K\rho_2 + \beta\sqrt{\rho_2} \end{cases}$$

Infrastructure: small system



Infrastructural bandwidth improves system performance

Infrastructure: large system



Infrastructural bandwidth must grow with system size

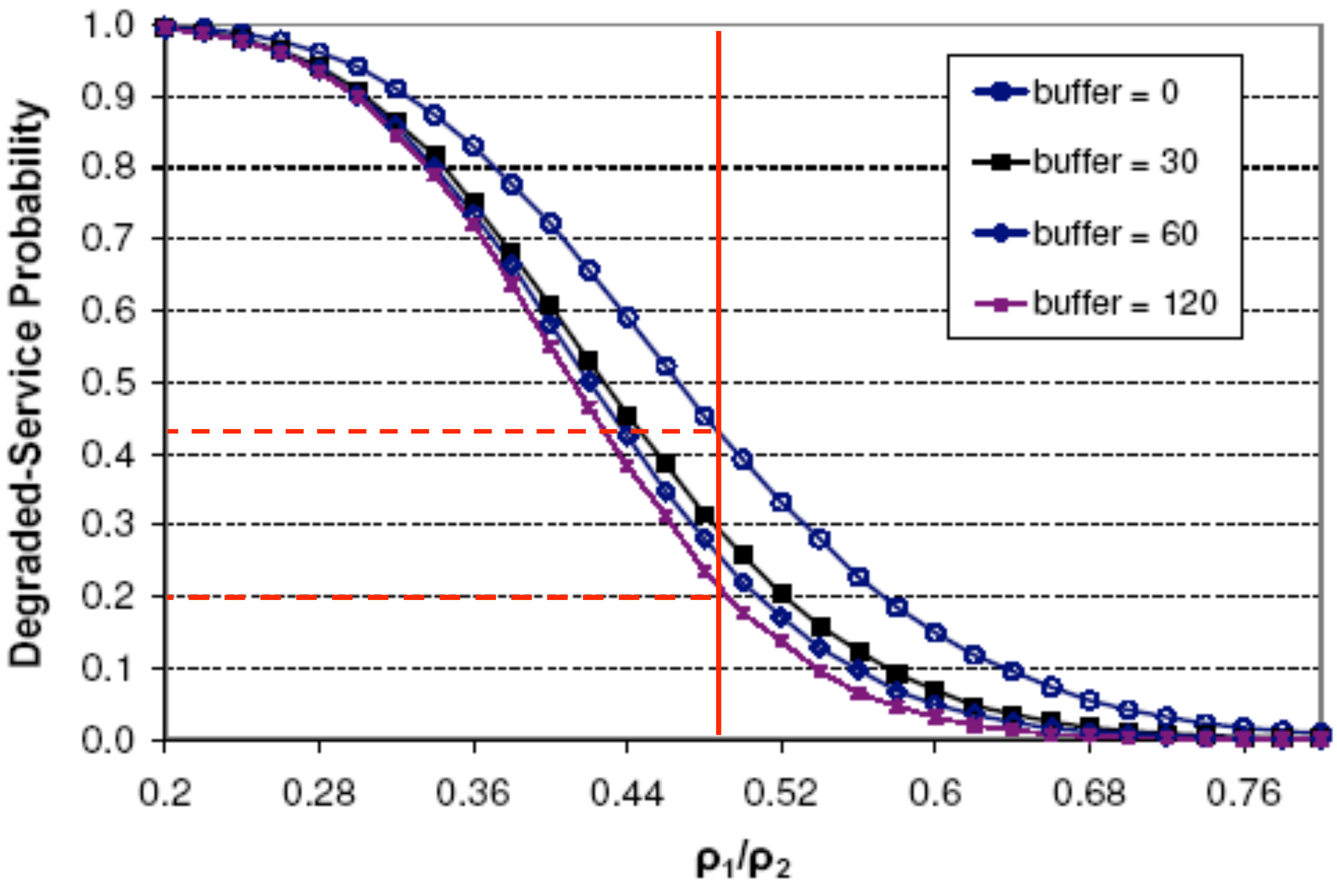
Buffering

- Peer churn causes fluctuations in a peer's download rate (from server and/or peers):

$$\phi(t) = \min\{u_s, \frac{u_s + u_1 L_1(t) + u_2 L_2(t)}{L_1(t) + L_2(t)}\}$$

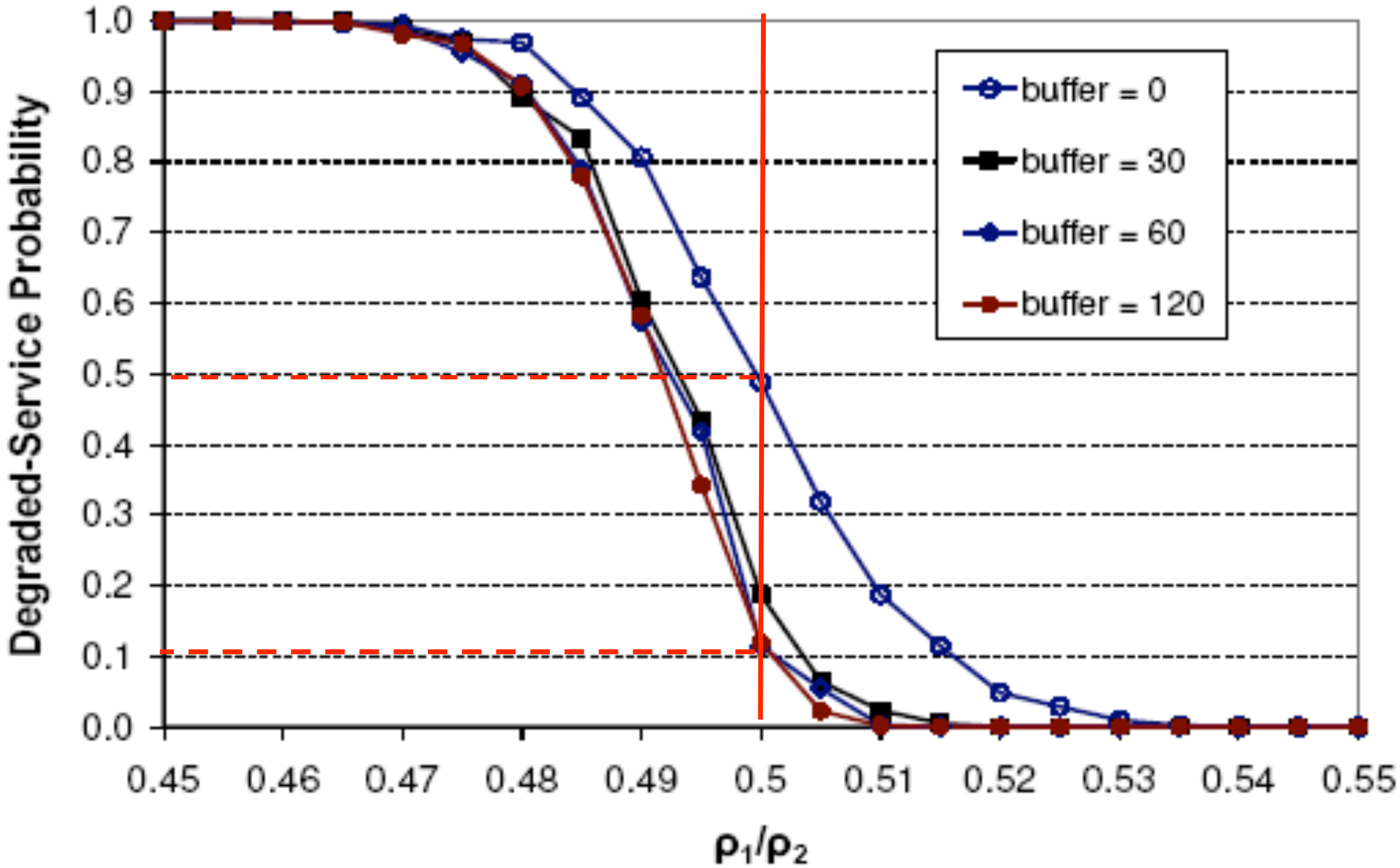
- Traditional streaming problem: bandwidth/delay fluctuations on client-server connections
  - solution: content buffering, delayed playback
- Pseudo-P2P-Live-Streaming
  - peers buffer d secs before playback
  - always download unfetched content at $I(t)$ from server/peers
  - skip content more than d secs old

Buffer Simulation: small system



Buffering improves performance dramatically.

Buffer Simulation: large system



(b) Large System $\lambda_2 = 10000$

More improvement for large systems

Lessons Learned

- Peer churn causes fluctuations in available bandwidth
  - "old days": network congestion if too many downloading clients
  - "p2p systems": bandwidth deficits if too few uploading peers
- Performance is largely determined by critical value
- Large systems have better performance
- Buffering can dramatically improve things
- Under-capacity region needs to be addressed
  - add more infrastructure
  - apply admission control and block ordinary peers
  - use scalable coding:
    - adapt transmission rate to available bandwidth
    - give lower rate to ordinary peers

Thanks!